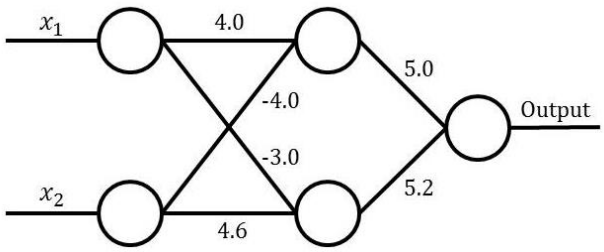
CSC 3520
Machine Learning
Florida Southern College

HW3: Neural Networks
Due: Tuesday, November 22, 2022

In this assignment, you will have the opportunity to:
- draw and analyze simple neural network architectures
- manually compute weight updates due to backpropagation
- write Python code to pass input data through a neural network
- demonstrate a conceptual understanding of decision boundaries learned by neural networks

1. **Backpropagation.** Consider the following 2-2-1 neural network with no bias:



The activation function for all hidden and output neurons in the network is the sigmoid function, $\sigma(a) = 1/(1 + e^{-a})$. Suppose you have one training example given by $(x_1, x_2) = (-0.5, 1)$ and a target $t = 0.9$.

(a) What is the output of the network?

(b) What is the mean squared error?

(c) Given the learning rate $\eta = 0.10$, what is the update (Δw) to the weight between the second hidden neuron and the output neuron (initially valued at 5.2)? Express your answer in scientific notation. You must show your work to earn full credit.

(d) What is the update to the weight between the first input neuron and the first hidden neuron (currently valued at 4.0)? Express your answer in scientific notation. You must show your work to earn full credit.

(e) If the target was changed to $t = 1$, would you expect the sign of the weight updates in (c) and (d) to change? Why or why not? *(HINT: You should not need to recompute the weight updates to answer this question.)*

2. **Visualizing Decision Boundaries.** Consider a 2-2-1 neural network with bias. Let the activation function at the hidden and output neurons be the hyperbolic tangent function, $\sigma(a) = \alpha \tanh \beta a$, where $\alpha = 1.716$ and $\beta = 2/3$. Suppose the matrices defining the input-to-hidden weights ($w^{(1)}$) and the hidden-to-output weights ($w^{(2)}$) are given as, respectively,

$$\begin{bmatrix} 0.5 & -0.5 \\ 0.3 & -0.4 \\ -0.1 & 1.0 \end{bmatrix} \text{ and } \begin{bmatrix} 1.0 \\ -2.0 \\ 0.5 \end{bmatrix}$$

where $w_{ij}^{(1)}$ connects the *i*th input neuron to the *j*th hidden neuron, and $w_{jk}^{(2)}$ connects the *j*th hidden neuron to the *k*th output neuron. The first row in all weight matrices corresponds to the bias.

(a) Sketch the network described above and label the neurons and weights accordingly.

(b) Write a Python script (`decisionboundaries.py`) to evaluate the network in the 2-dimensional input space defined by $\{(x_1, x_2) \in \mathbb{R}^2 \mid -5 \leq (x_1, x_2) \leq 5\}$. Use a resolution of 0.2; in other words, each dimension should vary from $-5{:}\ 0.2{:}\ 5$. This yields a total of 2601 (x_1, x_2) input points. Plot the results using a green 'o' for positive outputs and a red 'x' for negative outputs.

(c) Repeat (b) for the following weight matrices *(NOTE: this can be done in the same python file)*:

$$\begin{bmatrix} -1.0 & 1.0 \\ -0.5 & 1.5 \\ 1.5 & -0.5 \end{bmatrix} \text{ and } \begin{bmatrix} 0.5 \\ -1.0 \\ 1.0 \end{bmatrix}$$

BONUS: Considering your two plots, describe in a few sentences how you might design a neural network to encode the decision boundary given in the figure below. Sketch the network.